AMENDMENTS TO THE CLAIMS

Please amend claims 1, 11, 22, 23, 31, and 32, as follows.

Listing of Claims:

1. (CURRENTLY AMENDED) A method for operating (implementing) a secondary operating system on a processor in addition to a primary operating system, the method comprising the steps of:

switching from the primary operating system to the secondary operating system based on an interrupt call;

loading a secondary operating system driver (SOS driver) of the primary operating system and activating said secondary operating system driver for loading and controlling said secondary operating system, said secondary operating system driver comprising an interrupt handling routine;

exchanging interrupt tables of the operating systems during said interrupt call; and

determining information stored in an interrupt table of the secondary operating system with said interrupt handling routine, said information corresponding to a point in the secondary operating system at which said interrupt call is to be serviced, wherein a program code filed in a tunnel area of a memory associated with the processor changes between the primary operating system and the secondary operating system.

2. (ORIGINAL) Method according to claim 1, wherein the secondary operating system driver (SOS driver) subsequently loads the secondary operating system.

3. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein the secondary operating system driver loads the secondary operating system.

4. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein memory contexts (virtual operating areas) are created in the central processing unit (CPU).

5. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein a change between the operating systems takes place by means of the SOS driver of the primary operating system and the board support package (BSP).

6. (CANCELED)

7. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein the secondary operating system controls a change to the primary operating system.

8. (ORIGINAL) Method according to claim 7, wherein a change from the secondary operating system to the primary operating system takes place when the secondary operating system is idle (entry into the idle loop).

9. (ORIGINAL) Method according to claim 8, wherein a change from the secondary operating system to the primary operating system takes place through an instruction in the program sequence of the secondary operating system.

10. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein a change from the primary operating system to the secondary operating system takes place through an interrupt call.

11. (CURRENTLY AMENDED) Method according to claim 1, wherein the ~~change between operating systems takes place by means of a program code filed in~~ the tunnel area of the memory consists of a single memory page.

12. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein interrupt calls of the primary operating system are inhibited during the secondary operating system sequence.

13. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein an interrupt servicing routine in the SOS operator reads the interrupt call table of the secondary operating system and the processing of the latter takes place or is continued at the point relative to the interrupt call.

14. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein for each interrupt associated with the secondary operating system, which is to initiate an interrupt call in the secondary operating system, the system driver generates an entry in the interrupt call table in the primary operating system, which in turn initiates a call of the corresponding interrupt servicing routine in the secondary operating system.

15. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein by means of an interrupt call servicing routine in the system driver, the information stored in the interrupt table of the secondary operating system (SOS) is determined as to the point in the latter where the running of the interrupt is to take place.

16. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein in the case of activity of the secondary operating system (SOS) following an interrupt request through the information stored in the interrupt call table of the secondary operating system as to the point in the latter where the running of the interrupt is to take place, the interrupt call servicing routine of the secondary operating system (SOS) is directly polled solely by the secondary operating system and not via the system driver.

17. (PREVIOUSLY PRESENTED) Method according to claim 12, wherein after occurrence a corresponding interrupt call and determination of the point in the secondary operating system where interrupt running is to take place is determined, processing thereof at the point in the secondary operating system concerning the interrupt call is continued.

18. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein on changing from one operating system to the other all the system states of one operating system are stored.

19. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein on changing from one operating system to the other all system states of the other operating system are loaded.

20. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein clock generation for the secondary operating system takes place through the hardware timer.

21. (PREVIOUSLY PRESENTED) Method according to claim 1, wherein clock generation for the primary operating system takes place through a clock system driver.

22. (CURRENTLY AMENDED) A device for operating a secondary operating system on a processor in addition to a primary operating system, the device comprising:

a means for switching from the primary operating system to the secondary operating system based on an interrupt call;

a secondary operating system driver (SOS driver) of the primary operating system for loading and controlling the secondary operating system is constructed;

a function for exchanging interrupt tables of the operating systems, said secondary operating system driver comprising an interrupt service routine, said interrupt service routine determining information stored in an interrupt table of the secondary operating system corresponding to a point in the secondary operating system at which said interrupt call is to be serviced, wherein the SOS driver has a tunnel context setting routine for setting a tunnel context in the processor facilitating the change between operating systems by means of a program code filed in the tunnel context.

23. (CURRENTLY AMENDED) Device according to claim 22, wherein the ~~SOS drive has a tunnel context setting routine for setting a~~ tunnel context ~~in the central processing unit (CPU) the tunnel area of the memory~~ consists of a single memory page.

24. (CANCELED)

25. (PREVIOUSLY PRESENTED) Device according to claim 22, wherein the SOS driver has an interrupt call table change routine for producing entries in the interrupt call table of the primary operating system, which at least take up entries for the interrupt calls for the secondary operating system.

26. (PREVIOUSLY PRESENTED) Device according to claim 22, wherein the board support package (BSP) has a section for return to the primary operating system (POS).

27. (PREVIOUSLY PRESENTED) Device according to claim 22, wherein the secondary operating system driver (SOS driver) has an interrupt table section by means of which it produces in the primary operating system an interrupt call table containing a call of an interrupt servicing routine for polling the secondary operating system.

28. (PREVIOUSLY PRESENTED) Device according to claim 22, wherein the system driver is constructed for producing an entry in the interrupt call table in the primary operating system (POS) for each interrupt associated with the secondary operating system (SOS), which is intended to initiate an interrupt call in the secondary operating system and that the interrupt call table is constructed for polling the corresponding interrupt servicing routine in the secondary operating system (SOS).

29. (PREVIOUSLY PRESENTED) Device according to claim 22, wherein an interrupt call servicing routine in the system driver is constructed for determining the information stored in the SOS interrupt table as to the point in the secondary operating system where interrupt running is to take place.

30. (PREVIOUSLY PRESENTED) Device according to claim 22, wherein it is constructed in the case of activity of the secondary operating system (SOS) following an interrupt request through the information stored in the secondary operating system interrupt call table as to the point in which the secondary operating system interrupt running is to take place, so as to poll the interrupt call servicing routine of the secondary operating system directly solely through the secondary operating system and without passing via the system driver.

31. (CURRENTLY AMENDED) A method for operating (implementing) a secondary operating system on a processor in addition to a primary operating system in which a change from the primary operating system to the secondary operating system takes place through an interrupt call, the method comprising the steps of:

providing an interrupt call;

switching from the primary operating system to the secondary system based on said interrupt call <u>wherein a program code filed in a tunnel area of a memory associated with the processor switches between operating systems</u>;

providing the primary operating system with a secondary operating system driver, said secondary operating system driver comprising an interrupt call servicing routine;

loading said secondary operating system driver (SOS driver) in the primary operating system and activating said secondary operating system driver for loading and controlling the secondary operating system, said primary operating system having a primary operating system interrupt table, said secondary operating system having a secondary operating system interrupt table;

replacing said primary operating interrupt table with said second operating system interrupt table during said interrupt call;

processing information stored in said second operating system interrupt table with said secondary operating system driver such that said interrupt call servicing routine determines the information stored in the second operating system interrupt table as to the point in the secondary operating system where the running of the interrupt is to take place.

32. (CURRENTLY AMENDED) A device for operating a secondary operating system on a processor in addition to a primary operating system in which a change from the primary operating system to the secondary operating system takes place through an interrupt call, the device comprising:

a function for replacing an interrupt table of the primary operating system with an interrupt table of the secondary operating system a secondary operating system driver (SOS driver) of the primary operating system for loading and controlling the secondary operating system, said secondary operating system driver comprising an interrupt call servicing routine, wherein said interrupt call servicing routine determines the information stored in the interrupt table of the secondary operating system as to the point where interrupt running is to take place in said secondary operating system, wherein a program code filed in a tunnel area of a memory of the device and associated with the processor changes between operating systems.